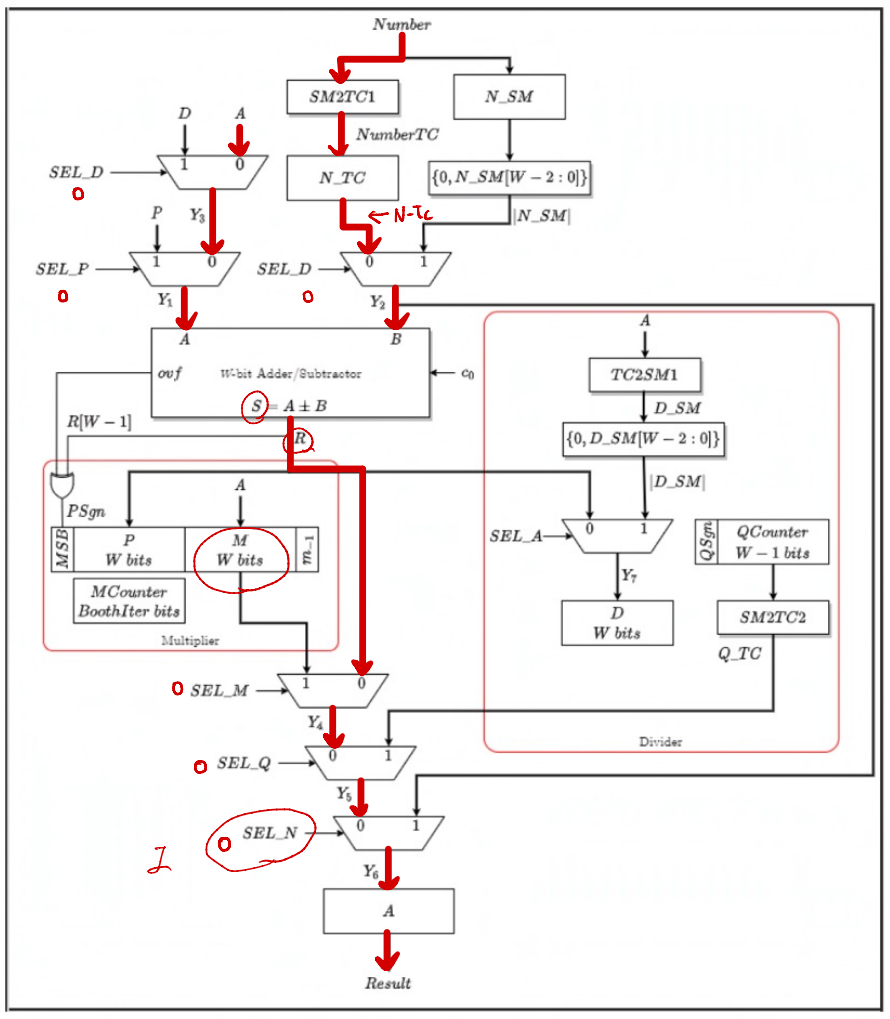
1. The Four Function Calculator datapath (Figure 3) can be simplified to do ONLY addition. Exam the data path and see what components and paths are necessary to accomplish. You should be able to set the relevant control signals to logic 1 or 0 to accomplish this. For example, the mux control signal that chooses the division data path is not required, so you can set SEL_Q = 0. Submit a copy of the datapath figure 3 anotating the control signal. *Hint: this is not much of a departure from the cash register example provided in lecture.* (10 points)



4. Provide the Verilog for your muxes using the labels in your simplified datapath. (10 points)

```
/------------------------------------------------------------------
/ MUXes
/ Use conditional assignments to create the various MUXes
/ following the example for MUX Y1
/------------------------------------------------------------------

  wire SEL_P;
  wire signed [W-1:0] Y1;
  assign Y1 = SEL_P? PM[WW:W+1] : Y3; // 1: Y1 = P; 0: Y1 = Y3

  //prelab

  wire SEL_N;
  wire signed [W-1:0] Y6;
  assign Y6 = SEL_N? N_TC : R;
```

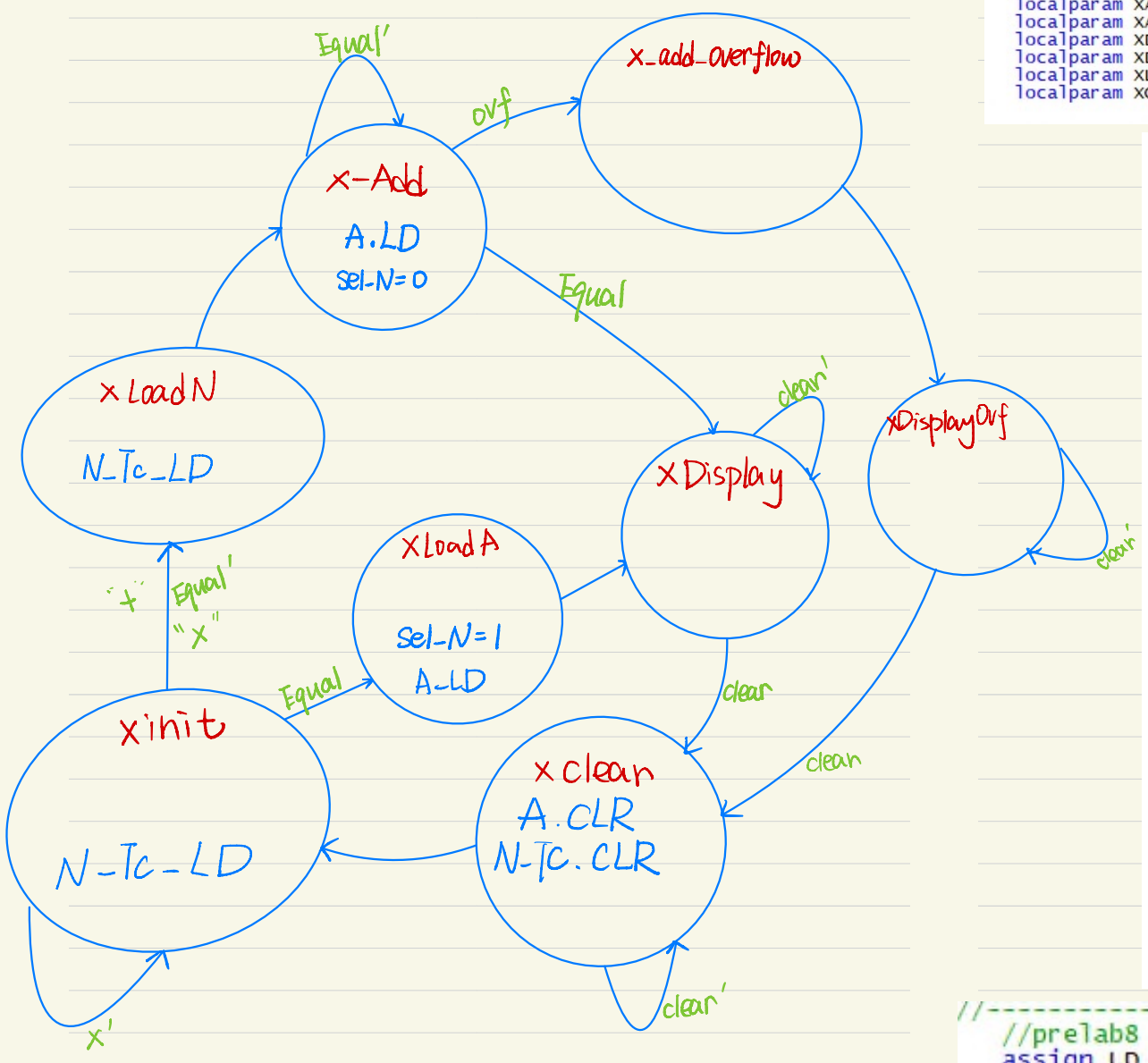
5. Provide the Next State and Ouput Verilog for your datapath controller using the labels in your simplified datapath. (10 points)

```
//------------------------------------------------------------------
// Controller State and State Labels
// Replace ? with the size of the state registers X and X_Next after
// you know how many controller states are needed.
// Use localparam declarations to assign labels to numeric states.
// Here are a few "common" states to get you started.
//------------------------------------------------------------------

  reg [?:0] X, X_Next;

  localparam XInit = 'd0; // Power-on state (A == 0)
  localparam XClear = 'd1;       // Pick numeric assignments
  localparam XLoadA = 'd2;
  localparam XResult = 'd3;
```

2. Provide the data path controller state diagram to do addition. You should provide data path control in terms using the control signals from your simplified datapath. *Hint: Refer to the high level state diagram Figure 5, expand and provide details to do addition and subtraction on the simplified data path.* (10 points).




```
//prelab 8
localparam XInit = 'd0;
localparam XLoadN = 'd1;
localparam XAdd = 'd2;
localparam XAddOvf = 'd3;
localparam XDisplayOvf = 'd4;
localparam XDisplay = 'd5;
localparam XLoadA = 'd6;
localparam XClear = 'd7;
```

```
      X_Next <= XInit;
//prelab8
XInit:
   if (Equals)
      X_Next <= XLoadA;
   else
      X_Next <= XLoadA;
XLoadN:
      X_Next <= XAdd;
XAdd:
   if (Equals)
      X_Next <= XDisplay;
   else if (Overflow)
      X_Next <= XAddOvf;
   else
      X_Next <= XAdd;

XAddOvf:
      X_Next <= XDisplayOvf;

XDisplayOvf:
   if (Clear)
      X_Next <= XClear;
   else
      X_Next <= XDisplayOvf;

XDisplay:
   if (Clear)
      X_Next <= XClear;
   else
      X_Next <= XDisplay;

XLoadA:
      X_Next <= XDisplay;

XClear:
      X_Next <= XInit;
```

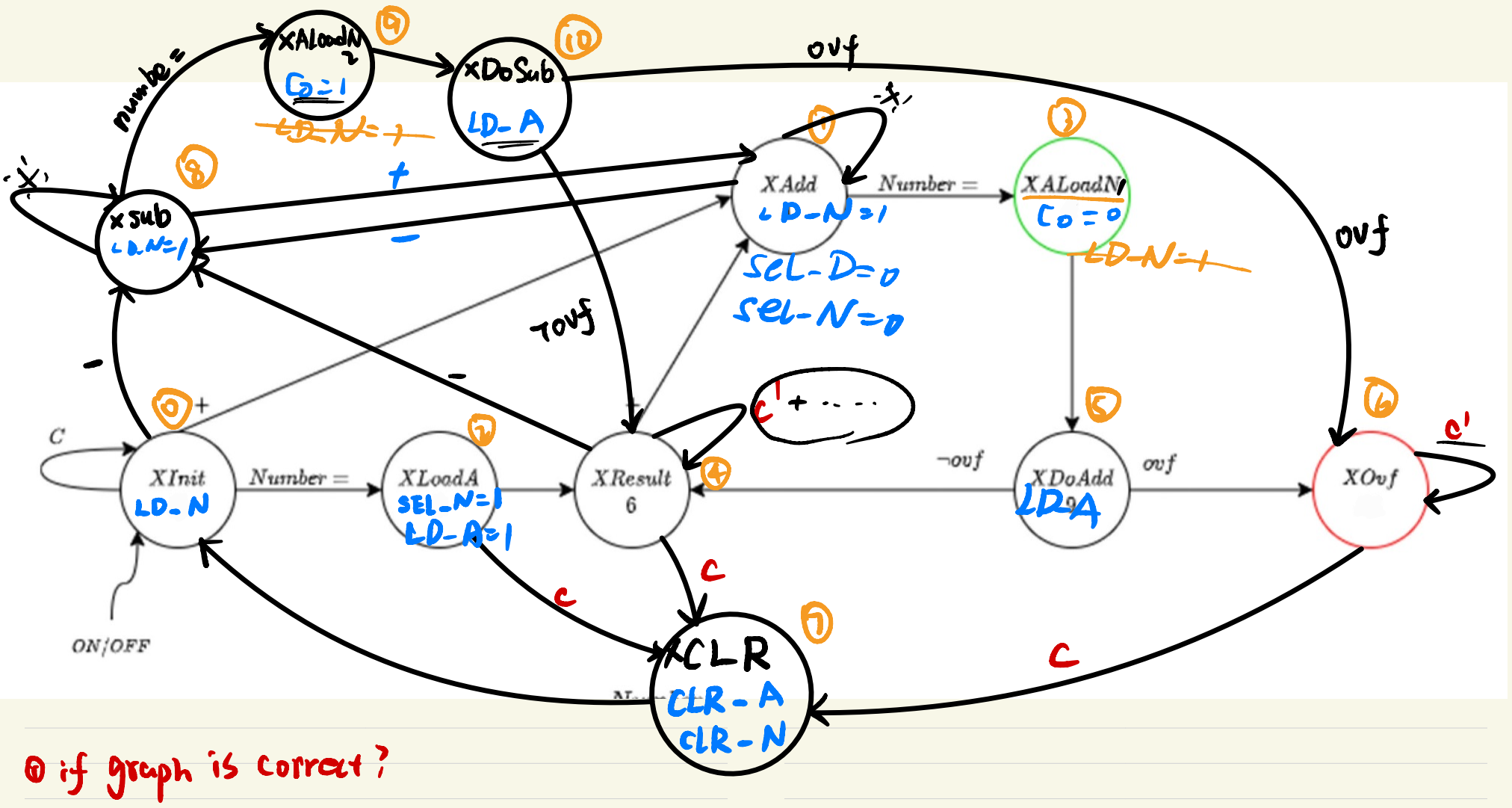
3. Provide the Verilog for your registers using the labels in your simplified datapath. (10 points)

```
// Registers
// For each register, declare it along with the controller commands that
// are used to update its state following the example for register A
//------------------------------------------------------------------

  reg signed [W-1:0] A;          // Accumulator
  wire CLR_A, LD_A;       // CLR_A: A <= 0; LD_A: A <= Q

  //prelab8
  reg signed [W-1:0] N_TC;
  wire CLR_N_TC, LD_N_TC;
```

```
//------------------------------------------------------------------
  //prelab8
  assign LD_A = (X == XLoadA || XAdd);
  assign CLR_A = (X == XClear);
  assign CLR_N_TC = (X == XClear);
  assign LD_N_TC = (X == XInit || X == XLoadN);
//------------------------------------------------------------------
```



① if graph is correct?

② what is controler for each register

A : LD_A CLR_A

N : LD-N , SEL-N , CLR_N

③ ? where should I creat "SEL-D" or I don't need it for inlab

④ how I can add subtraction state

⑤ for subtraction. should we use SEL-D? Because two's complement number

⑥ where is the add subtract module.

⑦ what is Sign funtonality.

⑧ what is reg[? : 0] x. x_Next means

⑨ why register can = directly to Y1 = A.

⑩ input ⓪ or 1 for coding

⑪ Name of the register (N) or N_TC_

⑫ else if and what should be write on (else)?

⑬ C0=1 or 0 where we write it. as controller! or ?

⑭ Initial begin ☆. we should adjust right? Like x ⇐ XInit

⑮ should we creat new wire for register? CLR_A CLR_N_TC
LD-A LD_N_TC

what about SEL-N ? is this also a controller for register N?

⑯ Howabout the "R" in Datapath state update.